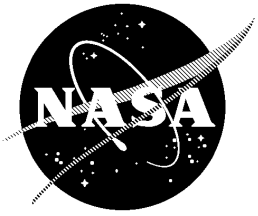
**Final Review**

**MISSION OPERATIONS AND DATA SYSTEMS DIRECTORATE**

# Landsat 7 Processing System (LPS) Software Configuration Guide

**July 1997**




National Aeronautics and Space Administration

Goddard Space Flight Center
Greenbelt, Maryland

**Final Review**

# Landsat 7 Processing System (LPS) Software Configuration Guide

**July 1997**

**Prepared by:**

Charlene Binley — Date
Software Configuration Manager

**Quality Assured by:**

Sheila Whisonant — Date
Quality Assurance Officer

**Concurred by:**

Khemraj Sharma — Date
Integration and Test Department Deputy Manager

**Approved by:**

Nate Daniel — Date
LPS Element Manager

Joy Henegar — Date
LPS Project Manager

**Goddard Space Flight Center**
**Greenbelt, Maryland**

# Abstract

This document describes the allocation of software and data to physical disks on the development and operational Landsat 7 Processing System (LPS). It also describes the software configuration management being applied to LPS, which includes procedural information for installing LPS and for building the LPS executables and installing them for testing and operations.

# Contents

## 1. Introduction



## 2. Physical and Logical Disk Assignments



## 3. Configuration Management

## List of Tables



## List of Figures

# Section 1. Introduction

## 1.1 Purpose

This software configuration guide is intended to provide in-depth procedural information for applying configuration management (CM) to the Landsat 7 Processing System (LPS) application software located at the customer site. This guide also explains how to build the LPS executables and where the new executables can be tested for verification.

Applicable documentation for this guide is listed in the References section.

## 1.2 Environments

LPS consists of five individual computers with identical operational environments. Each computer and its operational environment defines a string. Four of the strings support normal operations at all times. The fifth string is available for LPS test and maintenance support, as required, and as a backup string to the four operational strings. References 1 through 3 provide more specific information on the five LPS strings.

LPS provides a development environment on the fifth string for maintenance programmers who perform software enhancements and software corrections.

LPS also provides a test environment on the fifth string to test and verify changes to the LPS software before they are propagated to the four operational strings. The purpose and function of the test environment is further described in Reference 1.

LPS provides a CM environment on the fifth string to control access to the latest LPS operational software. LPS uses the Polytron Version Control System (PVCS) Version Manager as its CM tool.

## 1.3 Directory Structure for the Operational Environment

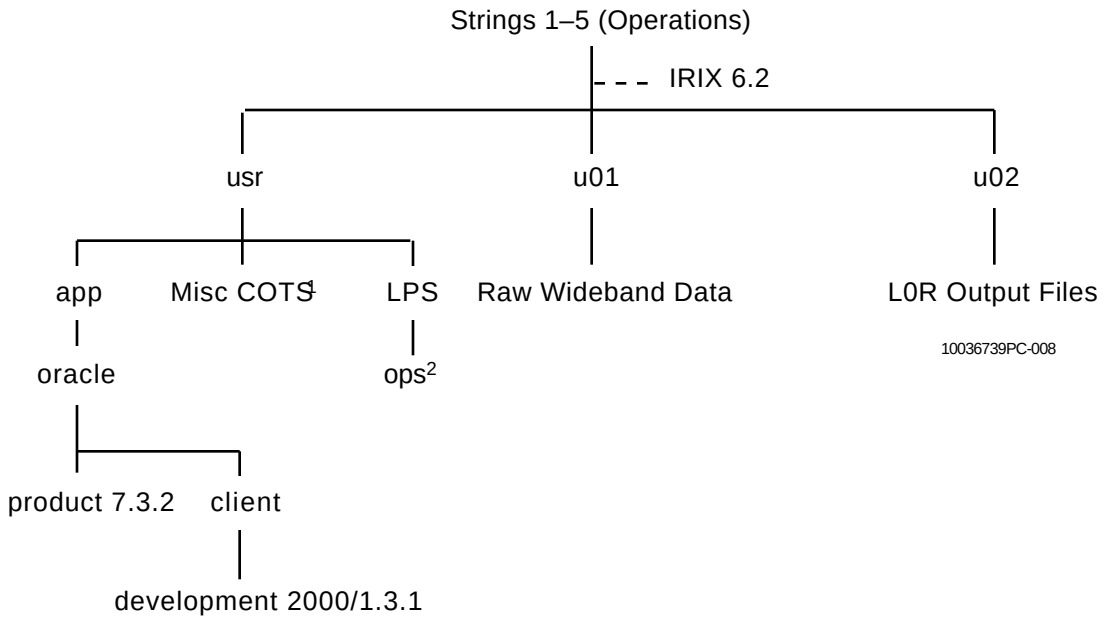
The directory structure of the operational environment (Figure 1–1) is defined in Reference 1. The four operational strings contain executable code and do not contain any application source files. New versions of the executables are generated and tested on the backup string and, if approved, promoted to the operational environment on the four LPS strings, as well as the operational directory on the fifth string.

## 1.4 Directory Structure for the Test Environment

The directory structure of the test environment is reflected in Figure 1–2.

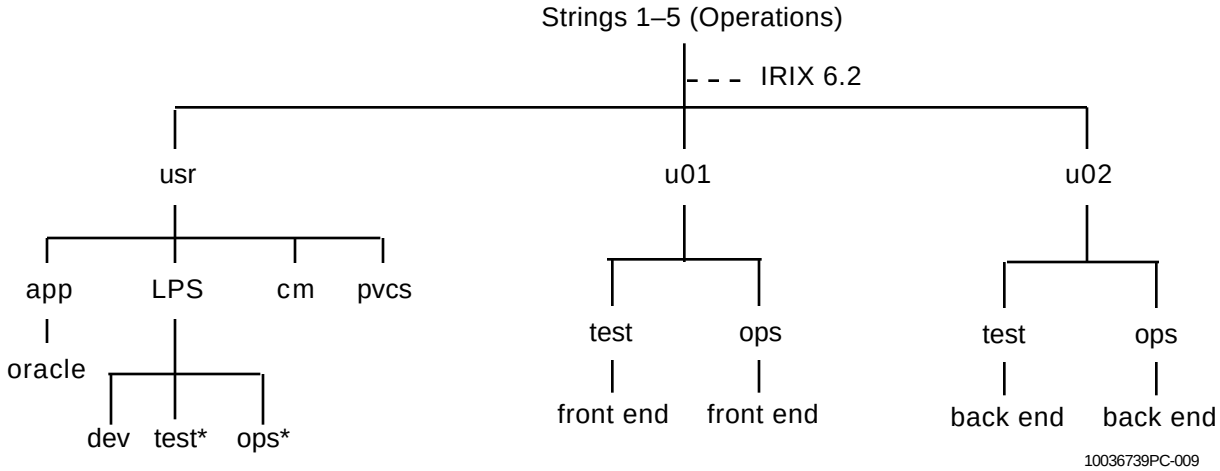
## 1.5 Directory Structure for the Development Environment

The directory structure of the development environment on string 5 stores developed software as shown in Figure 1–2. Source code (Rs) is stored in src and includes subdirectories; the executables (Rx) for the LPS applications are stored in the bin directory.




[1]Installation of the operating system, IRIX 6.2, automatically populates the miscellaneous COT
[2]Current version of the executables; see Figure 3–4 for the operational string structure.

***Figure 1–1. Operational Environment Directory Structure***

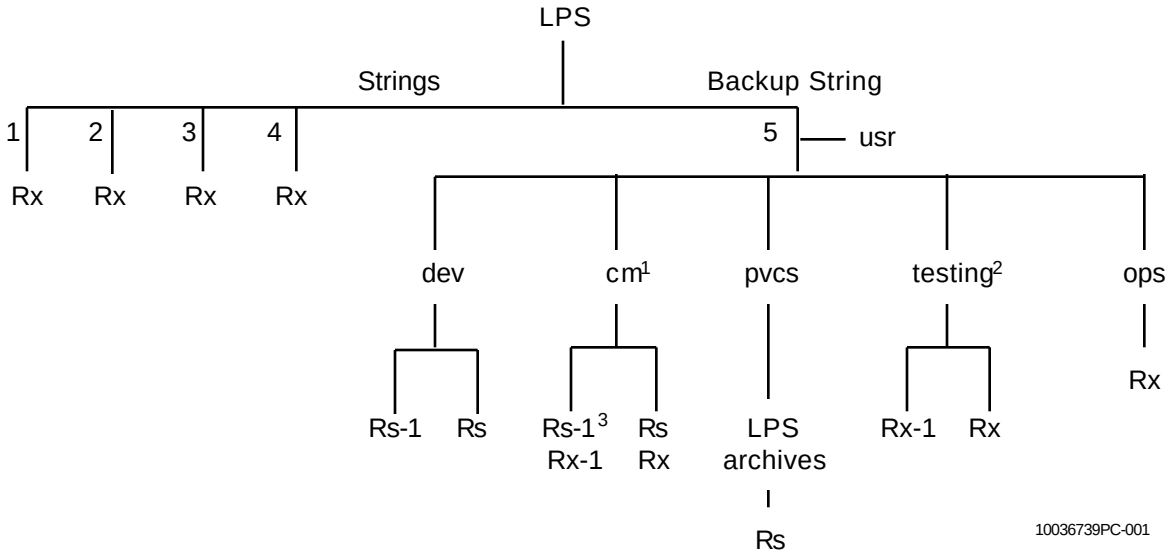



*See Figures 3–3 and 3–4 for more detail.

***Figure 1–2. String 5 – Test Environment Directory Structure***

### 1.5.1 Directory Structure for Source and Executables on Development String

The basic structure for storing developed software on string 5 is shown in Figure 1–3. Figures 3–2 and 3–3 show more detail of the directory structure. The "pvcs" subdirectory stores source code, and the "cm" subdirectory stores CM-related scripts and build LPS executables. The "ops" subdirectory houses LPS executables. Separate subdirectories for each build permit quick access to the current and previous source code. Once accepted, the LPS executable files are copied to the appropriate subdirectory on strings 1 to 4. Thus, the executables will be stored on all five strings.




Rs = current release's source for that environment

Rs-1 = previous release's source for that environment

Rx = current release's executables for that environment

Rx-1 = previous release's executables for that environment

[1]See Figure 3–1

[2]See Figure 3–2

[3]See Figure 3–3

***Figure 1–3. Basic Structure for Storing Developed Software***

# Section 2. Physical and Logical Disk Assignments

This section describes the physical disks on the operational and backup string disks and their logical allocation for the development, test, and operational environments.

## 2.1 Operational String Disks

Each operational string has a system disk and two Redundant Array of Inexpensive Devices (RAIDs), one for incoming data from the Landsat Ground Station (LGS), and a second RAID for temporary storage of LPS output data sets awaiting retrieval by the EOS Core System.

### 2.1.1 Logical-to-Physical Disk Mapping of Operational String

This section describes the logical disk allocation defined for the physical disks. Table 2–1 summarizes the allocation of the logical file systems to physical disks. This allocation is expected to meet operational needs.

***Table 2–1. Summary of Disk Allocation for Operational String***

| Disk | File System | Use | Allocation |
|---|---|---|---|
| System | /<br>/usr<br><br><br>/usr/LPS/ops | Operations | IRIX 6.2<br>ORACLE (Server and ORACLE Developer 2000 1.3.1)<br>Other commercial off-the-shelf (COTS) software (Section 2.1.1.1)<br>LPS executables |
| RAID 1 | /u01 | Front end | Raw wideband data |
| RAID 2 | /u02 | Back end | L0R output files |

#### 2.1.1.1 System Disk in Each Operational String

At any one time, the system disk on each operational string will contain the following:

Operating system and associated files

One operational instance of the LPS database, one for each environment (development, test, and operational)

COTS

- IRIX 6.2 (operating system)
- ONC3/NFS 6.2
- IRIS Development Option 6.2 (CaseVision Workshop 2.6)
- IRIX 6.2 Applications
- IRIS Power C 6.2

- ORACLE Server 7.3.2
- ORACLE Development 2000 1.3.1
- Network Time Protocol, Version 2.0 (SGI freeware)

The version numbers may change with each software turnover and will be noted in the turnover package.

#### 2.1.1.2 Redundant Array of Inexpensive Devices

##### 2.1.1.2.1 Input RAID

The input RAID on each operational string contains data received from the LGS that is ingested by the raw data capture subsystem (RDCS). The amount and frequency of this data are discussed in Reference 1.

##### 2.1.1.2.2 Output RAID

The output RAID on each operational string contains actual LPS output files. Users may request these files by notifying LPS. Output files are available to users via transfer protocol software.

## 2.2 Backup String Disks (String 5)

The backup string (string 5) has a system disk, an LPS development (or application) disk, and two RAIDs. The first RAID is reserved for storage of incoming data from LGS. As a development string, the second RAID is reserved for temporary storage of test data sets. As a backup to an operational string, the second RAID is reserved for temporary storage of LPS output data sets awaiting retrieval by the EOS Core System. Figure 2–1 shows the relationship of the backup string to the operational strings (strings 1 to 4).

### 2.2.1 Logical-to-Physical Disk Mapping of Backup String

This section describes the logical disk allocation defined for the physical disks. Table 2–2 summarizes the allocation of the logical file systems to physical disks.

***Table 2–2. Summary of Disk Allocation for Backup String***

| Disk | File System | Use | Software Allocation |
|---|---|---|---|
| System | /usr | Operations | Same as operational strings |
| LPS Application | /usr | Development | LPS software |
| RAID 1 | /u01 | Front end | Development: Test data only<br>As operational backup: Raw wideband data |
| RAID 2 | /u02 | Back end output | Development: Test data sets only<br>As operational backup: Level zero R output files |

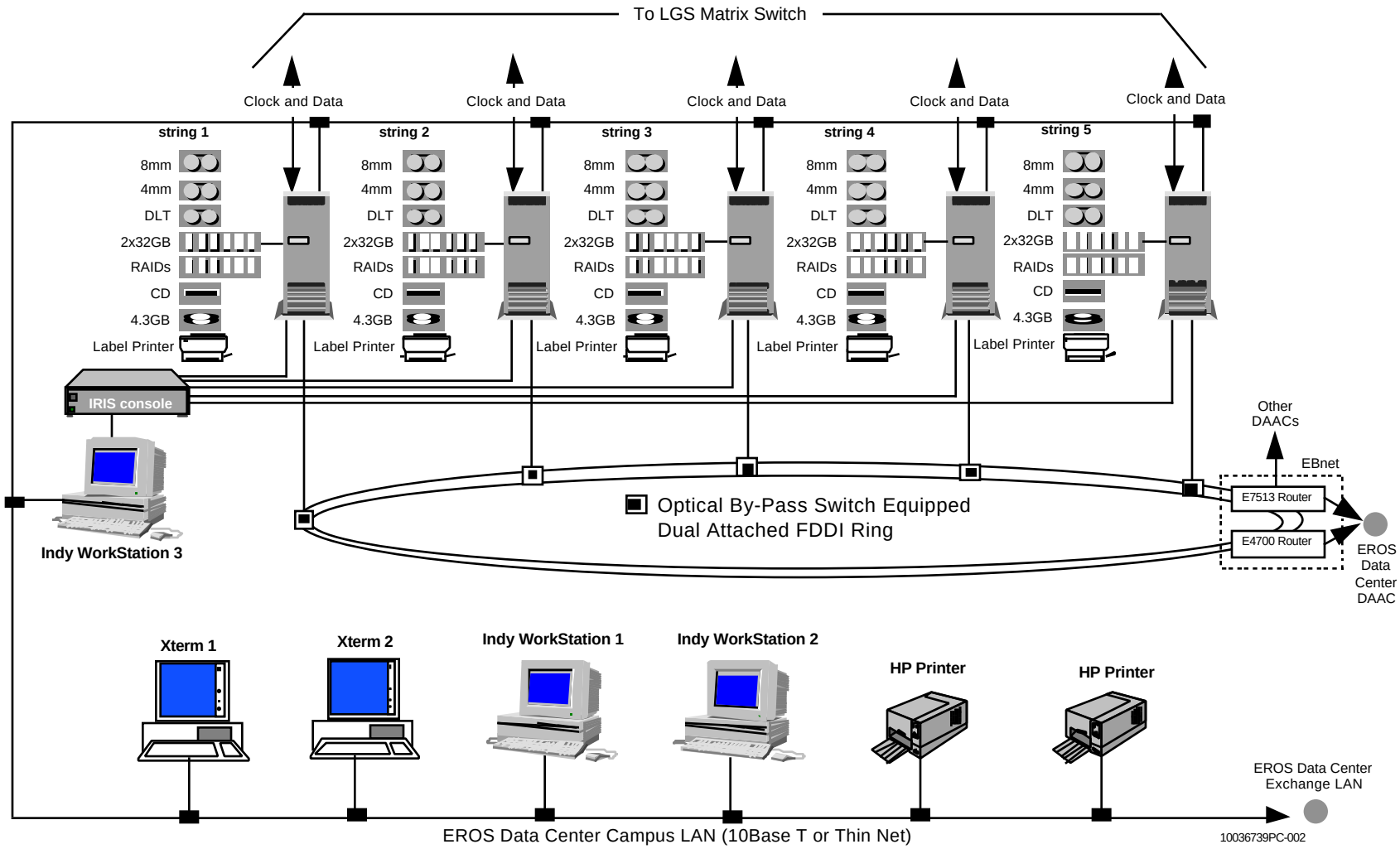



***Figure 2–1. Operational Hardware Configuration***

#### 2.2.1.1 System Disk (String 5)

At any one time, the system disk on string 5 will contain the following:

Operating system and associated files

Three distinct instances of the LPS database, one for each environment (development, test, and operational)

PVCS library

Development environment directories (see Section 1.3)

Government off-the-shelf (GOTS): frame_sync libraries

COTS

- Hierarchical data format (HDF) libraries
- IRIX 6.2 (operating system)
- NC3/NFS 6.2
- IRIS Development Option 6.2 (CaseVision Workshop 2.6)
- IRIX 6.2 Applications
- IRIS Power C 6.2
- ORACLE Server 7.3.2
- ORACLE Development 2000 1.3.1
- Network Time Protocol, Version 2.0 (SGI freeware)

#### 2.2.1.2 Redundant Array of Inexpensive Devices (String 5)

##### 2.2.1.2.1 Input RAID

At any one time, the input RAID contains raw data files used for testing

Image Assessment System (IAS) parameters from the calibration parameter file

Executables that have been modified to provide enhancements to LPS software or to correct errors

##### 2.2.1.2.2 Output RAID

At any one time, the output RAID contains

Test environment output data files resulting from IAS parameter, other L0R parameters or threshold tests

Development environment output data files resulting from tests of enhanced or modified software

Test databases (in each RAID) (TBD)

# Section 3. Configuration Management

This section discusses the CM directory structure and build process for LPS in the development string. All software written to support the operational LPS will be maintained and controlled by the configuration manager using the CM tool, PVCS Version Manager, which resides on the development string only. All builds must occur in a CM work directory that is outside of PVCS.

## 3.1 CM Directory Structure in the Development String

All LPS software builds must occur in the CM work directory called *lpswork*. All source and control files under configuration control are stored in the /pvcs/LPS/archives directory. Controlled files include script files and data files needed by certain subsystems. The CM directory structure is illustrated in Figure 3–1.

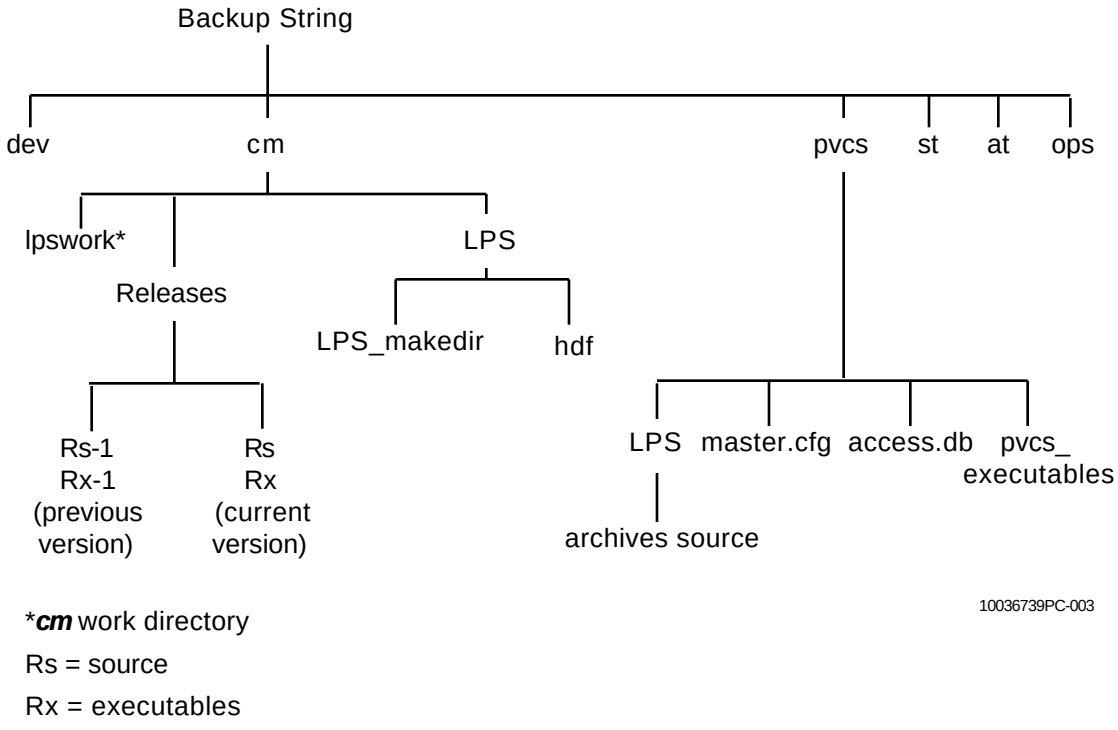



***Figure 3–1. CM Directory Structure in the Development String***

## 3.2 Building the System in the Development String

The software written to support LPS is developed, turned over to CM, stored in PVCS, extracted into the CM work directory, and then built for system testing. The executables are copied to operational directories only on approval of system test and operation management.

The process for building a version of LPS is always performed on the development string.

Scripts to build the LPS structure and manage PVCS activities exist and are listed in Section 3.2.1.3.

### 3.2.1 Preparation for a Build

Preparation for a build involves

Backup pvcs and current cm directories

Reviewing the turnover package

Checking that all COTS and GOTS software are the correct versions and are installed, including the required instances of the database in the test environment

Making necessary script changes to reflect LPS structural change in the CM work directory and PVCS

Checking the .cshrc and the .lpsdevrc files for correct environment setups; the .lpsrc is installed with the executables at the test site

Checking new and modified units into PVCS (Note: Keep a copy of the delivered software because the files are physically moved into PVCS.)

#### 3.2.1.1 Turnover Package

The turnover package contains all build information (a list of new, modified, and deleted units), environment changes, and build instructions.

#### 3.2.1.2 Required COTS and GOTS to Build

Section 2 identifies the COTS and GOTS software required for a build. The correct versions of the COTS and GOTS software should be noted in each build's turnover package. Some of them include the following:

IRIX 6.2

ORACLE 7.3.2

HDF4.Or1p1

HDF-EOSv1.00

Frame_Sync libraries

The usr/cm/LPS subdirectory should contain an hdf directory with appropriate versions of the HDF libraries so that a link can be made to it from the COTS subdirectory in the *usr/cm/lpswork/COTS* directory.

### 3.2.1.3 Script Changes

The following scripts may require changes if the LPS structure changes:

new_check: checks units into PVCS

new_get checks units out of PVCS into the work directory

LPS_makedir recreates the LPS working directory structure called "*lpswork*" where the build will take place. It also extracts the most current lmake and install scripts and places them in the appropriate subdirectories. The work directory is shown in Figure 3–2. (Table 3–1 provides an explanation of the terms used in Figure 3–2.)

setup_st_deliv creates the necessary subdirectory structure that will be populated to the test site. It should be updated as required; i.e., build number and other changes.

These scripts change when the LPS structure changes. All scripts should reflect the desired version label.

A copy of these scripts may need to be modified for patches or partial builds.

Any structural modifications to PVCS are made manually.

### 3.2.1.4 Changes to the LPS_makedir Script

The LPS_makedir script builds the complete LPS tree structure in the CM work directory, *lpswork*. It is a controlled script and requires a version label change whenever a new build takes place. LPS structural changes should also be made when necessary. Test the script before returning it to PVCS; this will require either checking in one copy of the modified lmake and install scripts into /usr/pvcs/LPS/archives with the correct version label or adding the correct version label to the current unchanged scripts stored in PVCS.

1. Extract the LPS_makedir script from PVCS:

   **cd /usr/cm/LPS**

   **get -l $PVCS_LPS/archives/LPS_makedir,v**

2. Change the version label (i.e., B3.1 or R1.0) and the necessary structural changes; B = build and R = release.

3. Save the changed file.

   If the lmake and install scripts are changed, check them into/usr/pvcs/LPS/archives.

   **put -vB# or R# /usr/cm/lpswork/install /usr/pvcs/LPS/archives/install,v**

   If the lmake and install scripts are unchanged, add the appropriate version label to each unit.

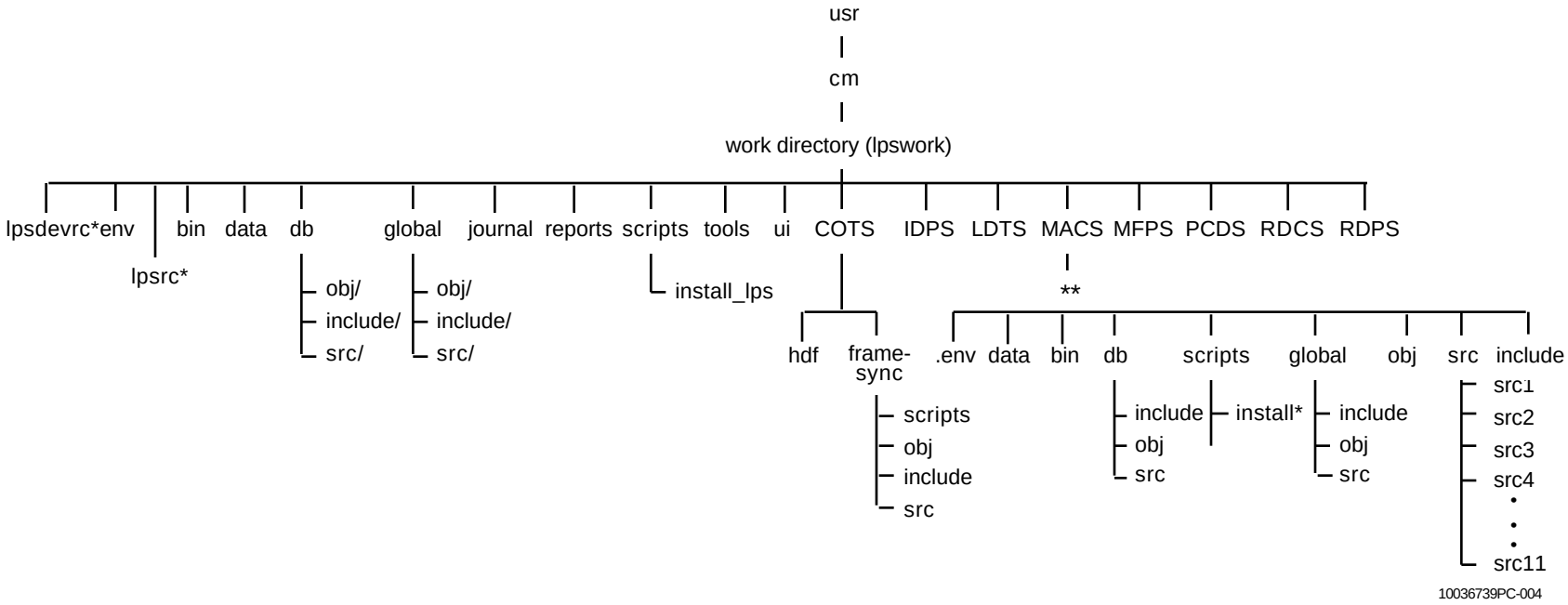
   Back up current lpswork directory if it is to be preserved or remove it:

**mv /usr/cm/lpswork bak.lpswork**

Or execute the LPS_makedir script to ensure directory structure reflects the correct LPS structure:

**cd /usr/cm/LPS**

Type **LPS_makedir**




**Each subsystem has all or some of this structure.

***Figure 3–2. Development Directory Structure***

***Table 3–1. Explanation of Terms in Figure 3–2***

| Term | Explanation |
|---|---|
| .env | Environment file that holds environment variables for the system and/or subsystem |
| bin | Holds executables |
| data | Holds data files; these directories may be subdivided further, perhaps for different contact periods |
| db | Holds global database routines and include files |
| global | Holds global routines and global include files; the global directory under the subsystem_name directory contains units common to the different subsystem processes, but not to the project as a whole |
| include | Holds include files |
| obj | Holds object files and db and global libraries |
| scripts | Holds make files and any other scripts |
| src | Holds source files |
| src# (# = 1–11) | Holds source files that create an executable; one src# for an executable |
| subsystem_name | Provides the top-level directory for each subsystem: IDPS, LTDS, MACS, MFPS, PCDS, RDCS, and RDPS |
| tools | Holds useful tools for developers and maintainers, such as alignments of code program and cadre tools |
| ui | Holds user interface files |
| tmp | Originally made available to hold temporary files (may be necessary for some systems); all files in this directory are strictly temporary and are deleted after every system reboot |

– Return the LPS_makedir to PVCS with the appropriate version label: **put -v[B# or R#] -m"/usr/cm/LPS/LPS_makedir /LPS/*archives*/LPS_makedir,v**

### 3.2.1.5 Verify User Environment Setup

The CM administrator must have the following statements in the CM .cshrc file to build in the development environment:

```
setenv LPS_HOME $HOME/lpswork

setenv HOME /usr/cm

setenv PVCSHOME /usr/pvcs

setenv PATH $PVCSHOME/.:$PATH

setenv PVCS_LPS /usr/pvcs/LPS

setenv INIT_HOME $HOME

setenv CMHOME $HOME/LPS
```

source .lpsdevrc (developers build environment variables)*

source .lpsrc [test (system test/acceptance test) build environment variables]*

Note: Only csh is supported.

Reference 4 documents the development environment variables for the LPS environment and for each subsystem's environment. Appendix D in Reference 1 describes the variables for the system test and the LPS operating environment.

#### 3.2.1.6 Check Modified and New Units into PVCS

Within the CM directory, all new and modified units are copied in the appropriate subdirectory of the work directory so they can be checked into PVCS. This includes any scripts involved, plus the LPS_makedir script, which will require at least a change in the version label for every build. Only one copy each of the lmake and install scripts is needed in PVCS and should be stored in the /usr/pvcs/LPS/archives directory. The other copies can be removed from the LPS structure. Also .a files are not checked into PVCS.

### 3.2.2 Detailed Build Process Using the Command Line Version

Once all preparations are complete as described in Section 3.2.1, follow the instructions in the turnover package to perform the following build activities:

Extract the LPS_makedir script to build the LPS structure tree.

Extract units from PVCS into the CM work directory for the appropriate release

Perform the build using the extracted units in the CM work directory.

Populate the test site(s).

The "new_get" script is executed to extract the appropriate units from PVCS and place them in the appropriate subdirectories in the CM work directory. When the LPS software is appropriately structured and the variables' paths are verified, the build script is executed.

Output reports are generated during a build and are stored in /usr/cm/reports for verification.

#### 3.2.2.1 Build the LPS Directory Structure

The following steps outline how to extract units from PVCS:

1. Login into the /usr/cm account.
2. Extract the LPS_makedir file from PVCS if not already extracted:

   **cd /usr/cm/LPS**

   **get vR2.0 [B#] -n $PVCS_LPS/archives/LPS_makedir,v**

3. Check that the file is an executable and change the mode it if needed:

**ls -la** (to see if file is an executable)

**chmod 775 LPS_makedir** (to make a file an executable)

4. Execute the LPS_makedir script to build the LPS work directory structure:

   **cd /usr/cm/LPS**

   Type **LPS_makedir**

5. When completed, verify that the LPS structure is correct.

#### 3.2.2.2 Extract Units From PVCs

1. Extract the appropriate units from PVCS using the "new_get" script.

   NOTE: The new_get script used should have been edited to reflect the correct R#/B# to extract the appropriate units.

   Type **new_get**

   Verify that the correct units have been extracted.

2. Validate the HDF link between /usr/cm/LPS/hdf and/usr/cm/lpswork/COTS/hdf; it is in the LPS_makedir script if needed.

   **ls -la /usr/cm/lpswork/COTS**

   You should see this line in the list: /usr/cm/lpswork/COTS/hdf --> /usr/cm/LPS/hdf

3. Validate the other environment variables for $HOME, $LPS_HOME, $INIT_HOME.

   **echo $INIT_HOME** (should be $HOME)

#### 3.2.2.3 Build Executables

Compile LPS as follows:

1. Go to scripts directory in the lpswork structure: cd $LPS_HOME_SCRIPTS
2. Create a script file to record the build; type **script filename.tscr**
3. Execute the build script: **install_lps nodebug**
4. Once the build is complete, type "exit" to discontinue recording the build
5. Check the
   - Executables in /usr/cm/lpswork/bin against the list of executables in the turnover/delivery package
   - File (filename.tscr) for error messages: **MORE filename.tscr lgrep "ERROR:"** (other keywords to chech are "unable," "cannot," "can't", and "don't know")

– Build instructions for any special setups required for system test, such as data files or files required to be in the system test's $HOME account

#### 3.2.2.4 Populate to Test Site

1. Prepare the structure that will be populated to the test site.
2. A script called setup_st_deliv is available to prepare the directory structure that is populated to the test sites (see Appendix C).
   – Edit the script. Change the build number "b#" in the line "mkdir b#_4_st" and add or delete any new subdirectories or special files as needed.
   – In the $HOME directory, execute the setup_st_deliv script. Figure 3–3 shows the directory structure of a populated test environment.
3. Copy the R#_4_st directory to the test site (i.e., /usr/LPS/st):

   **cp -pr R#_4_st <test directory path>** (-p preserves the date)
4. Edit the .cshrc file and set the LPS_HOME path to the appropriate build or release number:

   setenv LPS_HOME /usr/LPS/st/b#
5. Edit the .lpsrc file:

   Find and change any reference to b# to the appropriate build number (b#).

   Change the ORACLE_HOME path in the .lpsrc file:

   **setenv ORACLE_HOME /usr/app/oracle/client/developer2000/1.3.1**
6. Make sure that testers know to move the Ipd_mwd and Tk2Motif files to their HOME directory.
7. Verify that the environment variables and paths are correct before testing begins:

   **cd b# and source .lpsrc**
8. Once all verification is complete, change file ownership:

   **chown -R lpsst.user ***

   **chown lpsst.st .* (for the .lpsrc file)**
9. Have the system administrator change ownership of three RDCS executables and set user ID as follows:

   **chown root:sys rdc_Capture rdc_Transmit rdc_DeleteFiles**

   **chmod u+s rdc_Capture rdc_Transmit rdc_DeleteFiles**

Testing can begin.

### 3.2.3 Building for Operations

Because all source code is kept in PVCS on the backup string (string 5), the software builds for the operational environment are performed in the CM work directory on the backup string. When the build is complete and accepted by the testers and operations management, the executables are transferred to the operational site. The operational directory structure is shown in Figure 3–4.

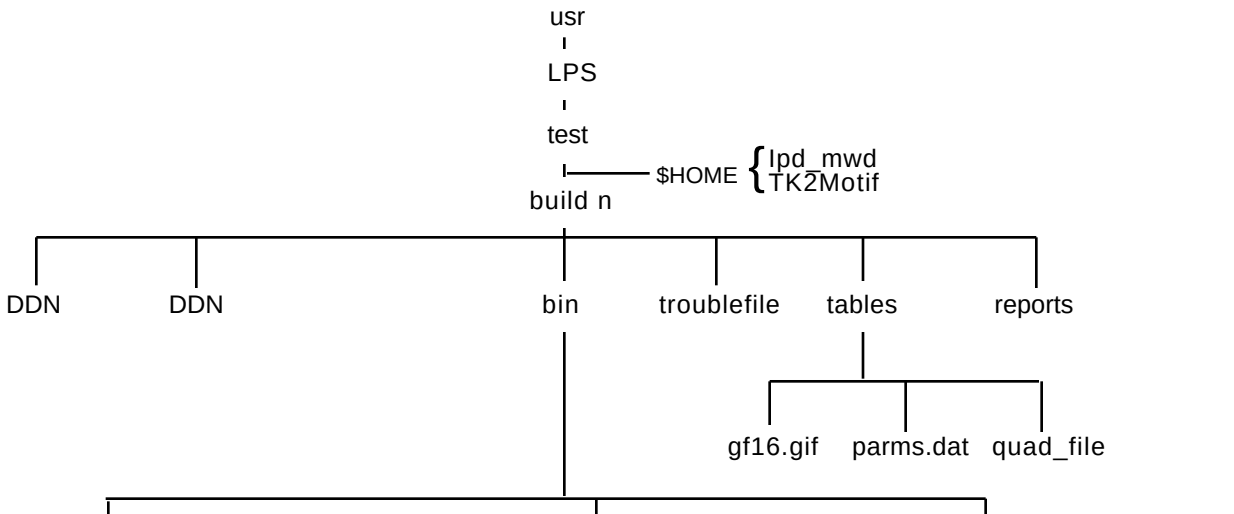


***Figure 3–3. Development Test Directory Structure***

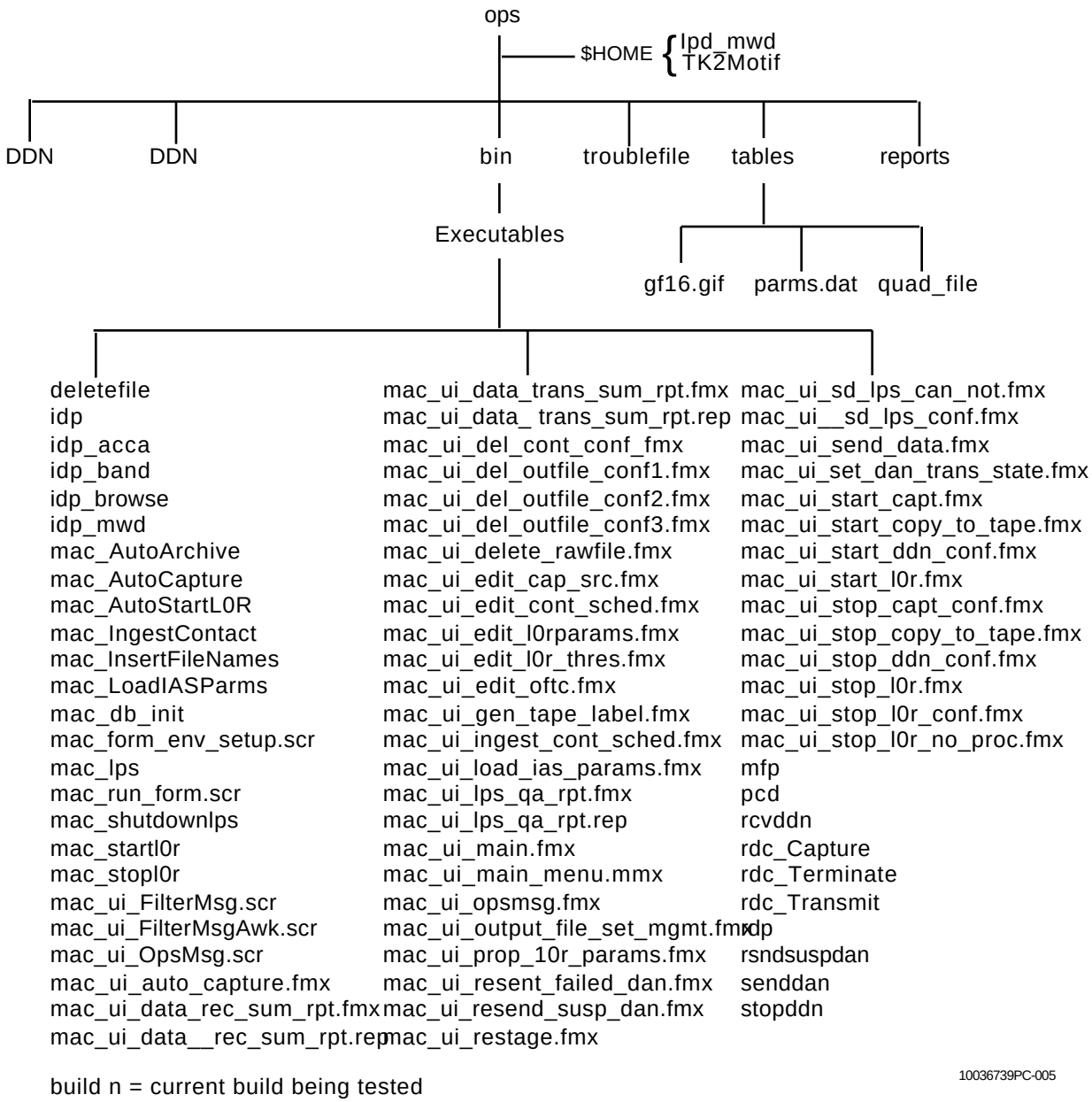


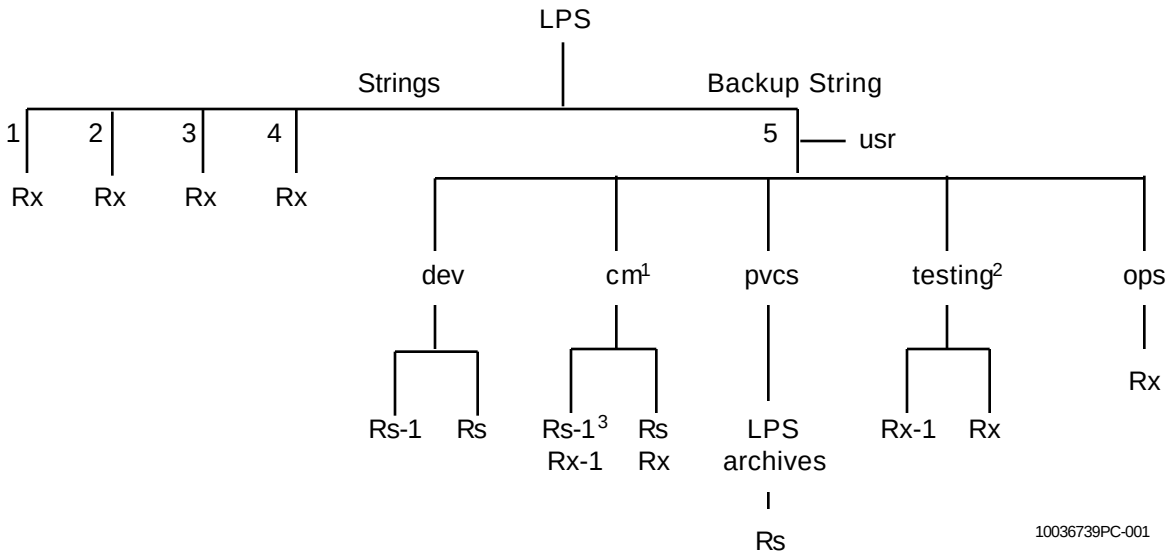

***Figure 3–4. Operations Directory Structure***

## 3.3 Populating the Runtime Directories

This section discusses the process of populating the runtime directories for the development system and the operational system once a build has been completed.

### 3.3.1 Development System

A runtime environment directory structure (Figure 3–5) exists on all LPS development hardware elements. This structure is identical for each element and contains identical directory structures for development, CM, and system test/acceptance test (Figure 3–3). The process for populating the runtime directories is the same for each test group as described in Section 3.2.2.4.




Rs = current release's source for that environment

Rs-1 = previous release's source for that environment

Rx = current release's executables for that environment

Rx-1 = previous release's executables for that environment

[1]See Figure 3–1

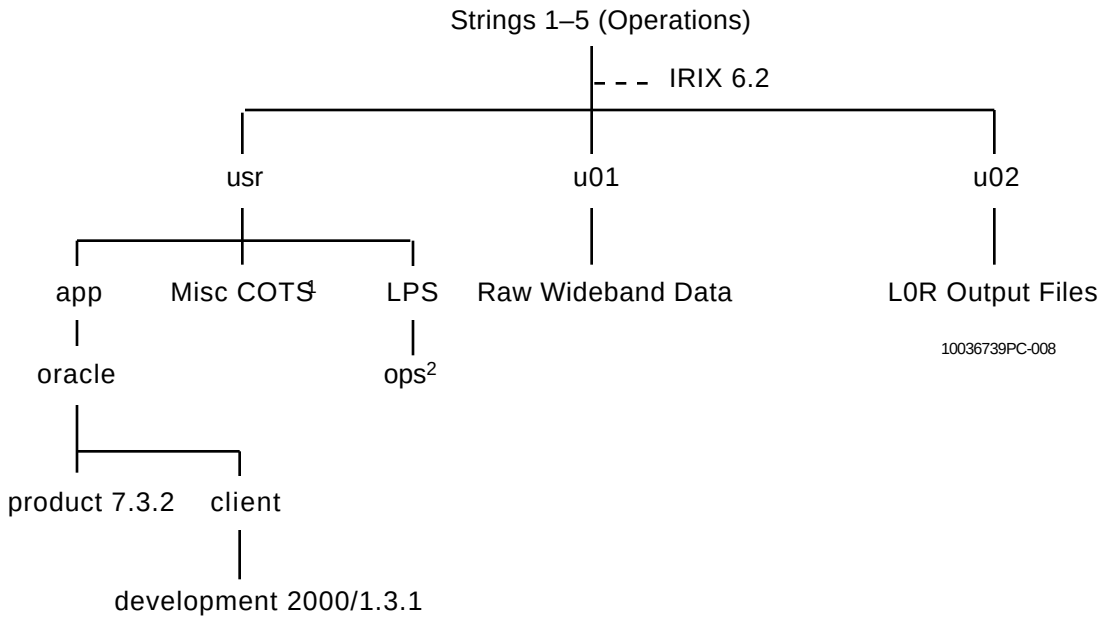
[2]See Figure 3–2

[3]See Figure 3–3

***Figure 3–5. Development Runtime Directory Structure***

### 3.3.2 Operational System

A build script, setup_ops, is available to create runtime directory structures on all operational hardware elements for the current release if those structures do not exist. The executables are copied or transferred via FTP to the /usr/LPS/ops directory on each operational string, where they will be available to each operational hardware element. The structure is the shown in Figure 3–6.




[1]Installation of the operating system, IRIX 6.2, automatically populates the miscellaneous COT

[2]Current version of the executables; see Figure 3–4 for the operational string structure.

***Figure 3–6. Operational Runtime Directory Structure***

Once the system is accepted by test and operations, the executables are populated to the operational strings. The executables are to be populated via FTP to strings 1–5 and copied to string 5, /usr/LPS/ops.

To populate to strings 1–4:

Prepare to send the build/release via FTP to another machine:

in lpsdev1 (string 5),

cd /usr/LPS/st/b#

TAR: tar cvf <b#R#ops.tar filename > < b#r#>& outfilename

Compress: Compress <tar file> (becomes .Z file)

In another screen, login to the target operational machine:

telnet lps00#

cd /usr/LPS/ops

ftp lpsdev1

cd /usr/LPS/st

get R#tarfile.Z

bye

Uncompress the .Z file.

Untar the .tar file: tar xvf b#r#.tar >$ b#r#tar.log.

Verify the executables and environment variables.

## 3.4 LPS Release Types

There are four types of LPS releases:

1. **Release** – Involves major functionality changes, a complete system rebuild, and goes through a complete testing cycle. Planned well in advance.
2. **Mini-Release** – Considered a maintenance release, and involves minor functionality changes. It can require either a full or partial rebuild, and goes through a complete testing cycle. Planned as a follow-on to a release.
3. **Patch** – Considered a means to provide critical functionality or correct mission-critical problems when sufficient time exists for a complete testing cycle. Involves a partial system rebuild.
4. **Engineering Mode** – Considered a means to provide mission-critical functionality when there is no time for a complete testing cycle. Only limited module, integration, and system testing is performed. Executables only are provided to expedite resolution of the problem. Source code and fill testing of the functionality is provided in the next scheduled release, mini-release, or patch of the system.

# Appendix A. Checklist for LPS Software Installation

☐ Use backup tape to restore system from full backup tape

OR

☐ 1. Install IRIS 6.2 Operating System (see Appendix B)*

☐ 2. Install Network Time Protocol

☐ 3. Set up directory structure (see Figure 1–1)

☐ 4. Install COTS (see Section 2)*

- ☐ Includes ORACLE Server (see Appendix C).

☐ 5. Install LPS executables

☐ 6. Verify environment variables and paths (see Appendix D)

☐ 7. Run LPS

*Refer to vendor installation procedures.

# Appendix B. Installing the IRIX Operating System

Follow the vendor's installation instructions. Make the following modifications after installation is complete (you must be "root" to make them):

1. Switch user to root:

   **su Root**

2. Edit /etc/syslog.conf:

   a. Add the following line at the end of the file:

      **add local0.debug;kern.none /tmp/LPS_Journal**

   b. Comment out:

      **comment out local0.alert root, operator**

   c. Save the file.

   d. Issue the **killall -HUP syslogd** command.

3. Run systune interactively by typing **systune -i**:

   a. Change semmns to 2048

   b. Change semmni to 100

   c. Quit and reboot

# Appendix C. Installing ORACLE

## C.1 Installing the ORACLE Server

The ORACLE Server (V7.3.2) was installed following the builder's installation instructions. The following responses are entered at the installer prompts.

1. Before running the installer, create the server directory:

   **mkdir /usr/app/oracle/product/7.3.2**

2. Set the environment variables as follows:

   **ORACLE_HOME=/usr/app/oracle/product/7.3.2**

   **ORACLE_SID=LPS**

   **ORACLE_TERM=iris**

   **LD_LIBRARY_PATH=$ORACLE_HOME/lib**

   **PATH=$ORACLE_HOME/bin:/usr/local/bin:$PATH**

   **ORACLE_PATH=$PATH**

3. Run the installer:

   **cd /CDROM/orainst**

   **./orainst**

4. Enter the following information at the prompts:

   Select the Installer activity:

   **Install, Upgrade, or De-Install Software**

   Select the Installer option:

   **Install New Product**

   Enter mount point for your software installation: /usr

   **OK**

   Complete $ORACLE_HOME location: /usr/app/oracle/product/7.3.2

   **OK**

   Do you want to create DB objects also?

   **Yes**

   Confirm (or change) log file location:

SQL Log: /usr/app/oracle/product/7.3.2/orainst/sql.log

Makefile Log: /usr/app/oracle/product/7.3.2/orainst/make.log

OS Log: /usr/app/oracle/product/7.3.2/orainst/os.log

**OK**

For the README file /CDROM/orainst/README.FIRST:

**Do not display this README file in the future**

Select one of the following:

**Install from CD-ROM**

Enter your ORACLE_SID: LPS

**OK**

Select the native language to be installed:

**American/English**

Do you want to relink ORACLE product executables?

**Yes**

Post-installation steps that need to be run by root will be written to /usr/app/oracle/product/7.3.2/orainst/root.sh.

**OK**

Install online help for

**All Products Being Installed**

Do you want to install UNIX-specific documentation?

**No**

Install product documentation from the product documentation CD-ROM for

**No Products**

Select the following products from the product list:

Advanced Replication Option

IRIX Specific Enhancements

ORACLE Intelligent Agent 7

ORACLE Names 2.0.1.1.0

ORACLE Server Manager (Motif)

ORACLE7 Distributed Database option 7.3.2.1.0

ORACLE7 Server (RDBMS) 7.3.2.1.0

ORACLE7 XA Library 1.1.1.0.

PL/SQL V2 2.3.2.0.0

Pro*C 2.2.2.0.0

SQL*Module 1.1.4.0.0

SQL*Net (V2) 2.3.2.1.0

SQL*Plus 3.3.2.0.0

TCP/IP Protocol Adapter (V2)

**Install**

Enter the official hostname (including domain) for this server: lpsdev1.gsfc.nasa.gov

**OK**

Enter a TCP service port for the Administration Server: 8888

**OK**

Enter passwd for administration.

Select group to act as DBA of the database:

**dba**

Select the OSOPER group: dba

**OK**

Choose storage type for database:

**Filesystem-Based Database**

Distribute control files over three mount points?

**Yes**

Enter /usr/app/oracle/ for all three mount points.

Select the character set for creating this database:

**US7ASCII**

Enter passwords for SYSTEM and SYS.

Do you want to set the passwords for the internal users (dba and operator)?

**No**

Would you like MTS (multi-threaded server) configured and the SQL*Net listener automatically started?

**No**

Do you want to use the following files as control files?

/usr/app/oracle/oradata/LPS/control01.ctl

usr/app/oracle/oradata/LPS/control02.ctl

/usr/app/oracle/oradata/LPS/control03.ctl

**Yes**

The following are the default file names and sizes that will be used to create the new database. Select OK to continue to the next screen of defaults files.

Default file names and sizes continued. Select Back to see the previous screen of defaults. Select OK to continue.

**OK**

Select Yes to accept the default file names and sizes shown on the previous screens. Select No to specify new values. Select Back to view the previous screens.

**Yes**

You've selected to install the IRIX-specific enhancements.

**OK**

Would you like to load the SQL*Plus Help Facility?

**Yes**

Would you like to load the SQL*Plus Demo Tables?

**Yes**

Please enter the directory where the X Windows libraries (libXt, libX11) may be found: /usr/lib

**OK**

Installation of shared Oracle library for Pro*C, OCI, and XA clients is complete.

**OK**

The requested action has been performed for selected products. Select OK to continue.

**OK**

5. Exit from the installer.
6. Run the root.sh script as the root.

## C.2 Creating the LPS Logical Database

1. Set ORACLE_HOME and ORACLE_SID:

   **ORACLE_HOME=/usr/app/oracle/product/7.3.2**

   **ORACLE_SID=LPS**

2. Follow the steps indicated in the db_creation_notes script.

## C.3 Creating the LPS Database

### C.3.1 Assumptions

1. Instance has been created and is up and running.
2. No users exist.
3. Environment variables ORACLE_HOME and ORACLE_SID are set.

### C.3.2 Procedures

1. Logon as system.
2. Create the tablespace(s):

   **SQL> @create_appl_ts.sql**

   **SQL> select * from dba_tablespaces;**

3. Create and grant quotas for the application account owner (appldba) and grant connect role to account owner:

   **SQL> @create_appldba_user.sql**

   **SQL> select * from all_users;**

   **SQL> connect appldba**

   **SQL> select * from session_privs;**

   **SQL> connect system**

4. Create the role(s):

   **SQL> @create_role.sql**

   **SQL> select * from dba_roles;**

5. Create all other users:

   **SQL> @create_inteam_user.sql**

   **SQL> select * from all_users;**

   **SQL> connect OPS$INTEAM/inteam**

   **SQL> select * from session_privs;**

   **SQL> connect system**

6. Grant the role(s) to users:

   **SQL> @grant_role.sql**

   **SQL> select * from dba_role_privs;**

7. Connect as the account owner (appldba).

8. Create the tables and sequences:

   **SQL> @create_table.sql**

   **SQL> select * from tab;**

   **SQL> @create_sequence.sql**

   **SQL> select * from seq;**

9. Generate the table and sequence privileges grants creation script:

   **SQL> @grant_privs_to_roles.sql**

   **SQL>!more tmp_grants.sql**

10. Generate the public synonyms creation script:

    **SQL> @create_public_syn.sql**

    **SQL>!more tmp_pubsyn.sql**

11. Grant the table and sequence privileges to the roles:

    **SQL> @tmp_grants.sql**

    **SQL>connect system**

    **SQL>set linesize 120**

**SQL>set pages 100**

**SQL>column privilege format a10**

**SQL>select TABLE_NAME,PRIVILEGE,GRANTEE,OWNER from dba_tab_privs**

**> where GRANTOR='APPLDBA'**

**> order by TABLE_NAME,PRIVILEGE;**

12. Connect as system and create the public synonyms:

**SQL> @tmp_pubsyn.sql**

**SQL>select SYNONYM_NAME , TABLE_NAME from dba_synonyms**

**> where OWNER = 'PUBLIC' and TABLE_OWNER = 'APPLDBA';**

**SQL>connect ops$inteam**

**SQL>desc PCD_MJF_ACCT;**

13. Connect as account owner (appldba) and load the static data:

**SQL> @load_static_data.sql**

**SQL>select * from LPS_Configuration;**

**SQL>select * from Valid_CCSDS_Parms;**

**SQL>select * from valid_ldt_parms;**

**SQL>select * from Valid_RDP_Thres;**

14. Log out and load the rest of the static data for valid_wrs_parms:

**pcd_sqlldr_command (this will use sql loader and data file pcd_data_file.dat, control file pcd.ctl and writes to the log pcd.log)**

**log in and check it:**

**% sqlplus appldba**

**SQL> select count(*) from valid_wrs_parms;**

## C.4 Installing ORACLE Developer 2000

The ORACLE Developer 2000 (V1.3.1) was installed following the builder's installation instructions. The following responses are entered at the installer prompts:

1. Before running the installer, create the *client* and *tns admin* directories:

**mkdir /usr/app/oracle/client/developer2000/1.3.1**

**mkdir /usr/app/oracle/client/developer2000/tns**

2. Set the environment variables as follows:

   **ORACLE_HOME=/usr/app/oracle/client/developer2000/1.3.1**

   **ORACLE_SID=LPS**

   **TNS_ADMIN=$ORACLE_HOME/tns**

   **ORACLE_TERM=iris**

   **LD_LIBRARY_PATH=$ORACLE_HOME/lib**

   **PATH=$ORACLE_HOME/bin:$PATH**

   **ORACLE_PATH=$PATH**

3. Run the installer:

   **cd /CDROM/orainst**

   **./orainst**

4. Enter the following information at the prompts:

   Select the Installer activity:

   **Install, Upgrade, or De-Install Software**

   Select the Installer option:

   **Install New Product**

   Enter $ORACLE_HOME location:

   **/usr/app/oracle/client/developer2000/1.3.1**

   Confirm (or change) log file location:

   SQL Log: /usr/app/oracle/client/developer2000/1.3.1/orainst

   Makefile Log: /usr/app/oracle/client/developer2000/1.3.1/orainst

   OS Log: /usr/app/oracle/client/developer2000/1.3.1/orainst

   **OK**

   For the README file /CDROM/orainst/README.FIRST:

   **Do not display this README file in the future**

   Select one of the following:

   **Install from CD-ROM**

Select the native language to be installed:

**American/English**

Do you want to relink ORACLE product executables?

**Yes**

Post-installation steps that need to be run by root will be written to /usr/app/oracle/client/developer2000/1.3.1/orainst/root.sh.

**OK**

Install online help for:

**All Products Being Installed**

Do you want to install UNIX-specific documentation?

**No**

Install product documentation from the product documentation CD-ROM for:

**OK**

Select the following products from the list:

ORACLE Forms 4.5.7.0.12

ORACLE Reports 2.5.5.1.1

PL/SQL Procedure Builder 1.5.6.12.1

SQL*Net (V2) 2.3.2.1.0

SQL*Plus 3.3.2.0.0

TCP/IP Protocol Adapter (V2) 2.3.2.1.0

**Install**

Would you like to install the PL/SQL Procedure Builder demos?

**No**

Please enter the directory where the X-Windows libraries (libXt, libX11) may be found: /usr/lib

**OK**

Please enter the directory where the MOTIF library (libXm) is located: /usr/lib

**OK**

Would you like to install the ORACLE Graphics 2.5 demos?

**No**

Select one or more user interfaces for ORACLE Forms:

**OK**

Would you like to install the ORACLE Forms demos?

**No**

Select one or more user interfaces for ORACLE Reports 2.5:

**Motif Bitmapped Interface**

Would you like to install the ORACLE Reports 2.5 demos?

**No**

Installation of shared ORACLE library for Pro*C, OCI, and XA clients is complete:

**OK**

The requested action has been performed for selected products. Select OK to continue:

**OK**

5. Exit from the installer.
6. Run the root.sh script as the root.

# Appendix D. Installing Network Time Protocol

To be supplied.

# Appendix E. Sample Environment Variables for CM

Note: The paths should be the same with the exception of /u04.

HOME=/u04/cm

LOGNAME=cm

HZ=100

TZ=EST5EDT

TERM=xterms

USER=cm

LANG=C

SHELL=/bin/tcsh

REMOTEHOST=lpscm.gsfc.nasa.gov

REMOTEUSER=UNKNOWN

MAIL=/usr/mail/cm

DISPLAY=198.119.37.190:0

SHLVL=1

PWD=/u04/cm

HOST=lpsdev1

HOSTTYPE=iris4d

MSGVERB=text:action

NOMSGLABEL=1

NOMSGSEVERITY=1

LD_LIBRARYN32_PATH=/usr/lib32

LD_LIBRARY64_PATH=/usr/lib64

LPS_HOME=/u04/cm/lpswork

INIT_HOME=/u04/cm

CM_HOME=/u04/cm/lpswork

PATH=.:/u04/cm/bin:/usr/pvcs/.:/u04/cm/lpswork/ui/bin:/u04/cm/lpswork/RDPS/bin:/u04/cm/lpswork/RDCS/bin:/u04/cm/lpswork/PCDS/bin:/u04/cm/lpswork/MFPS/bin:/u04/cm/lpswork/MACS/bin:/u04/cm/lpswork/LDTS/bin:/u04/cm/lpswork/IDPS/bin:/u04/cm/lpswork/tools/bin:/usr/pure/purif

y/:/u04/cm/lpswork/bin/:/usr/app/oracle/product/7.3.2/bin:/usr/bin/X11/:/usr/local/bin:/usr/bin:/bin:/usr/etc:/etc:/usr/bsd:/usr/sbin:/sbin:/u03/LPS/b3/COTS/hdf/hdfeos/bin/sgi

MANPATH=/usr/pure/purify/man:/u04/cm/lpswork/man:/usr/share/catman:/usr/share/man:/usr/catman:/usr/local/man

LD_LIBRARY_PATH=/usr/app/oracle/product/7.3.2/lib:/usr/lib

X11HOME=/usr/bin/X11

XNLSPATH=/usr/motif/lib/X11/nls

ORACLE_HOME=/usr/app/oracle/product/7.3.2

ORACLE_PATH=/usr/app/oracle/product/7.3.2/bin:/usr/bin/X11/:/usr/local/bin:/usr/bin:/bin:/usr/etc:/etc:/usr/bsd:/usr/sbin:/sbin

ORACLE_SID=LPS

ORACLE_TERM=xterm

TNS_ADMIN=/usr/app/oracle/client/developer2000/1.3.1/tns

TWO_TASK=lps

LPS_BIN=/u04/cm/lpswork/bin

LPS_DANFILE_PATH=/u04/cm/lpswork/DAN

LPS_DDNFILE_PATH=/u04/cm/lpswork/DDN

LPS_JOURNAL_PATH=/u04/cm/lpswork

LPS_OUTFILE_PATH=/u04/cm/lpswork/outfile

LPS_RAWFILE_PATH=/u04/cm/lpswork/rawfile

LPS_REPORT_PATH=/u04/cm/lpswork/reports

LPS_TAPE_DEV=/dev/rmt/tps131d5

LPS_TABLE_PATH=/u04/cm/lpswork/tables

LPS_TEMPFILE_PATH=/u04/cm/lpswork/tmp

LPS_TROUBLEFILE_PATH=/u04/cm/lpswork/troublefile

LPS_IAS_PARMS_PATH=/u04/cm/lpswork/iasparms

LPS_CONT_SCHED_PATH=/u04/cm/lpswork/schedules

LPS_PRINTER_DEVICE=/dev/plp

LPS_TAPE_LIBRARY_DEV=/dev/scsi/sc131d510

RDC_DEVICE=/dev/hpdiB

RDC_STATUS_INTERVAL=30

RDC_THRESH_SYSTEMDISK=0.01

LPS_CAPTURE_PROCESSOR=1

PURIFYHOME=/usr/pure/purify

HDF_HOME=/u04/cm/lpswork/COTS/hdf/4.0r2_IRIX_5.3

HDF_BIN=/u04/cm/lpswork/COTS/hdf/4.0r2_IRIX_5.3/bin

HDF_INC=/u04/cm/lpswork/COTS/hdf/4.0r2_IRIX_5.3/include

HDF_OBJ=/u04/cm/lpswork/COTS/hdf/4.0r2_IRIX_5.3/lib

HDF_SRC=/u04/cm/lpswork/COTS/hdf/4.0r2_IRIX_5.3/src

HDF_SCRIPTS=/u04/cm/lpswork/COTS/hdf/4.0r2_IRIX_5.3/scripts

HDF_EOS=/u04/cm/lpswork/COTS/hdf/hdfeos

HDFEOS_HOME=/u03/LPS/b3/COTS/hdf/hdfeos

HDFLIB=/u03/LPS/b3/COTS/hdf/4.0r2_IRIX_5.3/lib

HDFINC=/u03/LPS/b3/COTS/hdf/4.0r2_IRIX_5.3/include

MACHINE=SGI

OSTYPE=IRIX64

BRAND=sgi

NSL_FLAG=

CC=cc -32

F77=f77 -32

CFLAGS=-O -DIP19 -ansiposix

C_CFH=

CFHFLAGS=-O -DIP19 -ansiposix

C_F77_CFH=

C_F77_LIB=-lI77 -lU77 -lF77

F77FLAGS=

F77_CFH=

F77_C_CFH=

CFH_F77=

F77_C_LIB=

HDFSYS=IRIS4

HDFEOS_INC=/u03/LPS/b3/COTS/hdf/hdfeos/include

HDFEOS_BIN=/u03/LPS/b3/COTS/hdf/hdfeos/bin/sgi

HDFEOS_LIB=/u03/LPS/b3/COTS/hdf/hdfeos/lib/sgi

HDFEOS_OBJ=/u03/LPS/b3/COTS/hdf/hdfeos/obj/sgi

HDFEOS_SRC=/u03/LPS/b3/COTS/hdf/hdfeos/src

COTS_HOME=/u04/cm/lpswork/COTS

DB_HOME=/u04/cm/lpswork/db

DB_INC=/u04/cm/lpswork/db/include

DB_OBJ=/u04/cm/lpswork/db/obj

DB_SRC=/u04/cm/lpswork/db/src

FS_HOME=/u04/cm/lpswork/COTS/frame_sync

FS_INC=/u04/cm/lpswork/COTS/frame_sync/include

FS_OBJ=/u04/cm/lpswork/COTS/frame_sync/obj

FS_SCRIPTS=/u04/cm/lpswork/COTS/frame_sync/scripts

FS_SRC=/u04/cm/lpswork/COTS/frame_sync/src

GLOBAL_HOME=/u04/cm/lpswork/global

GLOBAL_INC=/u04/cm/lpswork/global/include

GLOBAL_OBJ=/u04/cm/lpswork/global/obj

GLOBAL_SRC=/u04/cm/lpswork/global/src

LPS_DATA=/u04/cm/lpswork/data

LPS_SCRIPTS=/u04/cm/lpswork/scripts

TOOLS_HOME=/u04/cm/lpswork/tools

TOOLS_BIN=/u04/cm/lpswork/tools/bin

TOOLS_INC=/u04/cm/lpswork/tools/include

TOOLS_OBJ=/u04/cm/lpswork/tools/obj

TOOLS_SRC=/u04/cm/lpswork/tools/src

UI_HOME=/u04/cm/lpswork/ui

UI_BIN=/u04/cm/lpswork/ui/bin

UI_INC=/u04/cm/lpswork/ui/include

UI_OBJ=/u04/cm/lpswork/ui/obj

UI_SRC=/u04/cm/lpswork/ui/src

IDPS_HOME=/u04/cm/lpswork/IDPS

IDPS_BIN=/u04/cm/lpswork/IDPS/bin

IDPS_DATA=/u04/cm/lpswork/IDPS/data

IDPS_DB=/u04/cm/lpswork/IDPS/db

IDPS_DB_INC=/u04/cm/lpswork/IDPS/db/include

IDPS_DB_OBJ=/u04/cm/lpswork/IDPS/db/obj

IDPS_DB_SRC=/u04/cm/lpswork/IDPS/db/src

IDPS_GLOBAL=/u04/cm/lpswork/IDPS/global

IDPS_GLOBAL_INC=/u04/cm/lpswork/IDPS/global/include

IDPS_GLOBAL_OBJ=/u04/cm/lpswork/IDPS/global/obj

IDPS_GLOBAL_SRC=/u04/cm/lpswork/IDPS/global/src

IDPS_INC=/u04/cm/lpswork/IDPS/include

IDPS_OBJ=/u04/cm/lpswork/IDPS/obj

IDPS_SCRIPTS=/u04/cm/lpswork/IDPS/scripts

IDPS_SRC=/u04/cm/lpswork/IDPS/src

LDTS_HOME=/u04/cm/lpswork/LDTS

LDTS_BIN=/u04/cm/lpswork/LDTS/bin

LDTS_DATA=/u04/cm/lpswork/LDTS/data

LDTS_DB=/u04/cm/lpswork/LDTS/db

LDTS_DB_INC=/u04/cm/lpswork/LDTS/db/include

LDTS_DB_OBJ=/u04/cm/lpswork/LDTS/db/obj

LDTS_DB_SRC=/u04/cm/lpswork/LDTS/db/src

LDTS_GLOBAL=/u04/cm/lpswork/LDTS/global

LDTS_GLOBAL_INC=/u04/cm/lpswork/LDTS/global/include

LDTS_GLOBAL_OBJ=/u04/cm/lpswork/LDTS/global/obj

LDTS_GLOBAL_SRC=/u04/cm/lpswork/LDTS/global/src

LDTS_INC=/u04/cm/lpswork/LDTS/include

LDTS_OBJ=/u04/cm/lpswork/LDTS/obj

LDTS_SCRIPTS=/u04/cm/lpswork/LDTS/scripts

LDTS_SRC=/u04/cm/lpswork/LDTS/src

MACS_HOME=/u04/cm/lpswork/MACS

MACS_BIN=/u04/cm/lpswork/MACS/bin

MACS_DATA=/u04/cm/lpswork/MACS/data

MACS_DB=/u04/cm/lpswork/MACS/db

MACS_DB_INC=/u04/cm/lpswork/MACS/db/include

MACS_DB_OBJ=/u04/cm/lpswork/MACS/db/obj

MACS_DB_SRC=/u04/cm/lpswork/MACS/db/src

MACS_GLOBAL=/u04/cm/lpswork/MACS/global

MACS_GLOBAL_INC=/u04/cm/lpswork/MACS/global/include

MACS_GLOBAL_OBJ=/u04/cm/lpswork/MACS/global/obj

MACS_GLOBAL_SRC=/u04/cm/lpswork/MACS/global/src

MACS_INC=/u04/cm/lpswork/MACS/include

MACS_OBJ=/u04/cm/lpswork/MACS/obj

MACS_SCRIPTS=/u04/cm/lpswork/MACS/scripts

MACS_SRC=/u04/cm/lpswork/MACS/src

MFPS_HOME=/u04/cm/lpswork/MFPS

MFPS_BIN=/u04/cm/lpswork/MFPS/bin

MFPS_DATA=/u04/cm/lpswork/MFPS/data

MFPS_DB=/u04/cm/lpswork/MFPS/db

MFPS_DB_INC=/u04/cm/lpswork/MFPS/db/include

MFPS_DB_OBJ=/u04/cm/lpswork/MFPS/db/obj

MFPS_DB_SRC=/u04/cm/lpswork/MFPS/db/src

MFPS_GLOBAL=/u04/cm/lpswork/MFPS/global

MFPS_GLOBAL_INC=/u04/cm/lpswork/MFPS/global/include

MFPS_GLOBAL_OBJ=/u04/cm/lpswork/MFPS/global/obj

MFPS_GLOBAL_SRC=/u04/cm/lpswork/MFPS/global/src

MFPS_INC=/u04/cm/lpswork/MFPS/include

MFPS_OBJ=/u04/cm/lpswork/MFPS/obj

MFPS_SCRIPTS=/u04/cm/lpswork/MFPS/scripts

MFPS_SRC=/u04/cm/lpswork/MFPS/src

PCDS_HOME=/u04/cm/lpswork/PCDS

PCDS_BIN=/u04/cm/lpswork/PCDS/bin

PCDS_DATA=/u04/cm/lpswork/PCDS/data

PCDS_DB=/u04/cm/lpswork/PCDS/db

PCDS_DB_INC=/u04/cm/lpswork/PCDS/db/include

PCDS_DB_OBJ=/u04/cm/lpswork/PCDS/db/obj

PCDS_DB_SRC=/u04/cm/lpswork/PCDS/db/src

PCDS_GLOBAL=/u04/cm/lpswork/PCDS/global

PCDS_GLOBAL_INC=/u04/cm/lpswork/PCDS/global/include

PCDS_GLOBAL_OBJ=/u04/cm/lpswork/PCDS/global/obj

PCDS_GLOBAL_SRC=/u04/cm/lpswork/PCDS/global/src

PCDS_INC=/u04/cm/lpswork/PCDS/include

PCDS_OBJ=/u04/cm/lpswork/PCDS/obj

PCDS_SCRIPTS=/u04/cm/lpswork/PCDS/scripts

PCDS_SRC=/u04/cm/lpswork/PCDS/src

RDCS_HOME=/u04/cm/lpswork/RDCS

RDCS_BIN=/u04/cm/lpswork/RDCS/bin

RDCS_DATA=/u04/cm/lpswork/RDCS/data

RDCS_DB=/u04/cm/lpswork/RDCS/db

RDCS_DB_INC=/u04/cm/lpswork/RDCS/db/include

RDCS_DB_OBJ=/u04/cm/lpswork/RDCS/db/obj

RDCS_DB_SRC=/u04/cm/lpswork/RDCS/db/src

RDCS_GLOBAL=/u04/cm/lpswork/RDCS/global

RDCS_GLOBAL_INC=/u04/cm/lpswork/RDCS/global/include

RDCS_GLOBAL_OBJ=/u04/cm/lpswork/RDCS/global/obj

RDCS_GLOBAL_SRC=/u04/cm/lpswork/RDCS/global/src

RDCS_INC=/u04/cm/lpswork/RDCS/include

RDCS_OBJ=/u04/cm/lpswork/RDCS/obj

RDCS_SCRIPTS=/u04/cm/lpswork/RDCS/scripts

RDCS_SRC=/u04/cm/lpswork/RDCS/src

RDPS_HOME=/u04/cm/lpswork/RDPS

RDPS_BIN=/u04/cm/lpswork/RDPS/bin

RDPS_DATA=/u04/cm/lpswork/RDPS/data

RDPS_DB=/u04/cm/lpswork/RDPS/db

RDPS_DB_INC=/u04/cm/lpswork/RDPS/db/include

RDPS_DB_OBJ=/u04/cm/lpswork/RDPS/db/obj

RDPS_DB_SRC=/u04/cm/lpswork/RDPS/db/src

RDPS_GLOBAL=/u04/cm/lpswork/RDPS/global

RDPS_GLOBAL_INC=/u04/cm/lpswork/RDPS/global/include

RDPS_GLOBAL_OBJ=/u04/cm/lpswork/RDPS/global/obj

RDPS_GLOBAL_SRC=/u04/cm/lpswork/RDPS/global/src

RDPS_INC=/u04/cm/lpswork/RDPS/include

RDPS_OBJ=/u04/cm/lpswork/RDPS/obj

RDPS_SCRIPTS=/u04/cm/lpswork/RDPS/scripts

RDPS_SRC=/u04/cm/lpswork/RDPS/src

CVINSTRLIB=/u04/cm

PVCSHOME=/usr/pvcs

PVCS_LPS=/usr/pvcs/LPS

CMHOME=/u04/cm/lpswork

DELIV=/u04/cm/Dev_Deliveries/b3.2del/b3

LOGS=/u04/cm/LOGS

ENV=/u04/cm/.kshrc

# Appendix F. Sample Environment Variables for System Test

Source .lpsrc to verify the environment for system test.

HOME=/export/home/lpsst

LOGNAME=lpsst

HZ=100

TZ=EST5EDT

TERM=iris-ansi-net

USER=lpsst

LANG=C

SHELL=/bin/tcsh

REMOTEHOST=lpsdev1.gsfc.nasa.gov

REMOTEUSER=UNKNOWN

MAIL=/usr/mail/lpsst

DISPLAY=lpscm.gsfc.nasa.gov:0.0

SHLVL=1

PWD=/export/home/lpsst

HOST=lps003

HOSTTYPE=iris4d

MSGVERB=text:action

NOMSGLABEL=1

NOMSGSEVERITY=1

LD_LIBRARYN32_PATH=/usr/lib32

LD_LIBRARY64_PATH=/usr/lib64

LPS_HOME=/usr/LPS/st/b3.2

HDF_HOME=/u01/hdf/HDF3.3r4/hdf

LPS_LOG_STDOUT=TRUE

PATH=.:/usr/home/lpsst/bin:/export/home/lpsst/tools:/usr/LPS/st/b3.2/bin/:/usr/app/oracle/client/developer2000/1.3.1/bin:/usr/bin/X11/:/usr/local/bin:/usr/bin:/bin:/usr/etc:/etc:/usr/bsd:/usr/sbin:/bin

MANPATH=/usr/LPS/st/b3.2/man:/usr/share/catman:/usr/share/man:/usr/catman:/usr/local/man

LD_LIBRARY_PATH=/usr/app/oracle/client/developer2000/1.3.1/lib:/usr/lib

X11HOME=/usr/bin/X11

XNLSPATH=/usr/motif/lib/X11/nls

ORACLE_HOME=/usr/app/oracle/client/developer2000/1.3.1

ORACLE_PATH=/usr/app/oracle/client/developer2000/1.3.1/bin:/usr/bin/X11/:/usr/local/bin:/usr/bin:/bin:/usr/etc:/etc:/usr/bsd:/usr/sbin:/sbin

ORACLE_SID=LPS

ORACLE_TERM=xterm

TNS_ADMIN=/usr/app/oracle/client/developer2000/1.3.1/tns

TWO_TASK=lps

LPS_BIN=/usr/LPS/st/b3.2/bin

LPS_DANFILE_PATH=/usr/LPS/st/b3.2/DAN

LPS_DDNFILE_PATH=/usr/LPS/st/b3.2/DDN

LPS_JOURNAL_PATH=/usr/LPS/

LPS_OUTFILE_PATH=/u02/st/b3.2/outfile

LPS_RAWFILE_PATH=/u01/st/b3.2/rawfile

LPS_REPORT_PATH=/usr/LPS/st/b3.2/reports

LPS_TAPE_DEV=/dev/rmt/tps131d2vc

LPS_TABLE_PATH=/usr/LPS/st/b3.2/tables

LPS_TEMPFILE_PATH=/u01/st/b3.2

LPS_TROUBLEFILE_PATH=/u02/st/b3.2/troublefile

LPS_IAS_PARMS_PATH=/usr/LPS/st/b3.2/iasparms

LPS_CONT_SCHED_PATH=/usr/LPS/st/b3.2/schedules

LPS_PRINTER_DEVICE=/dev/plp

LPS_TAPE_LIBRARY_DEV=/dev/scsi/sc131d2l0

RDC_DEVICE=/dev/hpdiB

RDC_STATUS_INTERVAL=30

RDC_THRESH_SYSTEMDISK=0.01

LPS_CAPTURE_PROCESSOR=1

ENV=/export/home/lpsst/.kshrc

# References

1. Computer Sciences Corporation, 514-3SUG/ 0195, *Landsat 7 Processing System (LPS) Users Guide for Release 1*, Draft, January 1997

2. —, 514-4BIP/0195, *Landsat 7 Processing System (LPS) Build Implementation Plan*, Revision 2, September 1996

3. —, 560-8SDS/0194, *Landsat 7 Processing System (LPS) Detailed Design Specification*, May 26, 1995

4. —, TBS, *Landsat 7 Processing System (LPS) Software Development Standards and Procedures*, date TBS

5. —, 514-30mm/0196, *Landsat 7 Processing System (LPS) Operations and Maintenance Manual*, Signature Copy, November 27, 1996

# Acronyms

| | |
|---|---|
| CM | configuration management |
| COTS | commercial off-the-shelf |
| FTP | File Transfer Protocol |
| GOTS | Government off-the-shelf |
| HDF | hierarchical data format |
| IAS | Image Assessment System |
| IDPS | image data processing subsystem |
| IT | integration test |
| LDTS | LPS data transfer subsystem |
| LGS | Landsat Ground Station |
| LPS | Landsat 7 Processing System |
| MFPS | major frame processing subsystem |
| PCDS | payload correction data subsystem |
| PVCS | Polytron Version Control System |
| RAID | Redundant Array of Inexpensive Devices |
| RDCS | raw data capture subsystem |
| TAR | tape archival retrieval |